Claims

1. (Previously amended) In a computer having a main memory, a method of enhancing scalability of server applications, comprising:

executing an application component under control of an operating service, the application component having a state and function code for performing work responsive to method invocations from a client;

providing an interface for the operating service to receive an indication from the application component that the work is complete;

maintaining the state in the main memory between the method invocations of the function code by the client in the absence of the indication from the application component that the work is complete; and

destroying the state of the application component in response to the indication from the application component to the operating service at the provided interface that the work is complete and without action by the client.

2. (Previously amended) The method of claim 1 wherein the operating service retains an operating service's reference to the application component and the step of destroying the state comprises releasing the operating service's reference to the application component by the operating service while the client retains a client's reference to the application component.

3. (Previously amended) The method of claim 1 wherein the step of destroying the state comprises resetting the state of the application component to the application component's initial post-creation state.

4. (Previously amended) The method of claim 1 wherein said destroying the state is performed by the operating service upon a next return of the application component from the client's call following the indication from the application component that the work is complete.

5. (Original) In a computer, a computer operating environment for scalable, component-based server applications, comprising:

a run-time service for executing an application component in a process, the application component having a state and implementing a set of functions;

an instance creation service operative, responsive to a request of a client, to return a reference to the application component through the run-time service to the client, whereby the client calls functions of the application component indirectly through the run-time service using the reference to initiate work by the application component; and

the run-time service being operative, responsive to an indication from the application component that the application component has completed the work for the client, to destroy the application component's state on the application component returning from a call by the client without action by the client.

7. (Original) The computer operating environment of claim 5 wherein the application component initiates the indication before returning from the call by the client, whereby the application component's state is destroyed immediately on return from the client's call without further action by the client.

8. (Once Amended) In a computer, a computer operating environment for scalable, component-based server applications, comprising:

a run-time service for executing an application component in a process, the application component having a state and implementing a set of functions;

an instance creation service operative, responsive to a request of a client, to return a reference to the application component through the run-time service to the client, whereby the client calls functions of the application component indirectly through the run-time service using the reference to initiate work by the application component;

the run-time service being operative, responsive to an indication from the application component that the application component has completed the work for the client, to destroy the application component's state on the application component returning from a call by the client without action by the client; and

a component context associated by the run-time service with the application component and providing an interface having a member function that the application component calls to initiate the indication.

9. (Original) The computer operating environment of claim 8 wherein the application component performs the work within a transaction and wherein calling the member function of the component context causes the transaction to abort.

10. (Original) The computer operating environment of claim 8 wherein the application component performs the work within a transaction and wherein calling the member function of the component context permits the transaction to commit.

11. (Original) The computer operating environment of claim 5 wherein the run-time service holds a reference to an instance of the application component, and destroys the application component's state by releasing the reference to the instance.

12. (Original) The computer operating environment of claim 5 wherein the run-time service destroys the application component's state by resetting the state.

13. (Previously amended) In a computer, a method of encapsulating state of processing work for a client by a server application in a component with improved scalability, comprising:

encapsulating function code and a processing state for the work in a component;

providing a reference through an operating service for a client to call the function code of the component to initiate processing of the work by the component;

receiving an indication from the component that the work by the component is complete; and

discarding the processing state of the component responsive to the component indicating completion of the work before receiving any indication from the client that the component's work is complete.

14. (Previously amended) In a computer, a method of encapsulating state of processing work for a client by a server application in a component with improved scalability, comprising:

encapsulating function code and a processing state for the work in a component;

providing a reference through an operating service for a client to call the function code of the component to initiate processing of the work by the component;

receiving an indication from the component that the work by the component is complete;

discarding the processing state of the component responsive to the component indicating completion of the work before receiving any indication from the client that the component's work is complete; and

performing the step of discarding the processing state upon a next return of the component from a call of the client following the indication from the component that the work is complete.

15. (Previously amended) In a computer, a method of encapsulating state of processing work for a client by a server application in a component with improved scalability, comprising:

encapsulating function code and a processing state for the work in a component;

providing a reference through an operating service for a client to call the function code of the component to initiate processing of the work by the component;

receiving an indication from the component that the work by the component is complete; and

discarding the processing state of the component responsive to the component indicating completion of the work before receiving any indication from the client that the component's work is complete;

wherein the step of receiving the indication includes providing a context object containing data representing a context of the component and having an integration interface for receiving a call of the component to indicate that the work by the component is complete.

16. (Original) The method of claim 15 wherein the call of the component to the integration interface of the context object further indicates that a transaction encompassing the work is to be committed.

17. (Original) The method of claim 15 wherein the call of the component to the integration interface of the context object further indicates that a transaction encompassing the work is to be aborted.

18. (Previously amended) In a computer, a system service for providing an execution environment for scalable application components, comprising:

code responsive to a request from a client program to create an application component for returning to the client program a reference through the system service to the application component;

code responsive to a call from the client program using the reference for initiating processing of work by the application component, the application component producing a processing state during processing the work;

code for receiving an indication from the application component that processing by the application component of the work is complete; and

code for destroying the processing state of the application component responsive to the indication from the application component that processing by the application component of the work is complete and without action from the client program.

19. (Original) The system service of claim 18 further comprising:

code for producing an instance of the application component and retaining a reference to the instance, the instance containing the processing state; and

wherein the code for destroying the processing state comprises code for releasing the reference to the instance without action from the client program to thereby cause the processing state to be destroyed.

20. (Original) The system service of claim 18 wherein the code for destroying the processing state comprises:

code for resetting the processing state to an initial state of the application component.

21. (Once Amended) In a computer having a main memory, a method of enhancing scalability of server applications, comprising:

executing an application component under control of an operating service, the application component having a state and function code for performing work responsive to method invocations from a client;

maintaining the state in the main memory between the method invocations of the function code by the client in the absence of an indication from the application component that the work is complete; and

destroying the state by the operating service in response to an indication from the application component without action by the client, such that the destroyed state is not persistent.

22. (Previously added) In a computer having a main memory, a method of enhancing scalability of server applications, comprising:

executing an application component under control of an operating service, the application component having a component data state and function code for performing work on a data resource responsive to method invocations from a client, the component's work on the data resource having a work data state, the component data state initially having an initial post-creation state upon the component's creation;

providing an interface for the operating service to receive an indication from the application component that the work is complete;

maintaining the component data state in the main memory between the method invocations of the function code by the client in the absence of the indication from the application component that the work is complete; and

in response to the indication and without client action, destroying the component data state by the operating service upon a next return of the application component from a method invocation of the client, while persistently maintaining the work data state.

23. (Previously added) The method of claim 22 wherein the step of destroying the component data state comprises resetting the component data state to the initial post-creation state.

24. (Previously amended) A computer readable program code-carrying media having software program code encoded thereon, the software program code for executing on a server computer and implementing a method of enhancing scalability of server applications, the method comprising:

executing an application component under control of an operating service, the application component having a component data state and function code for performing work on a data resource responsive to method invocations from a client, the component's work on the data resource having a work data state, the component data state initially having an initial post-creation state upon the component's creation;

providing an interface for the operating service to receive an indication from the application component that the work is complete;

maintaining the component data state in the main memory between the method invocations of the function code by the client in the absence of the indication from the application component that the work is complete; and

in response to the indication and without client action, destroying the component data state by the operating service upon a next return of the application component from a method invocation of the client, while persistently maintaining the work data state.

25. (Previously added) In a computer having a main memory, an application component having a state and function code for performing work responsive to method invocations from a client, an improved method of enhancing scalability of server applications, comprising:

providing an interface for the operating service to receive an indication from the application component that the work is complete;

maintaining the state in the main memory between the method invocations of the function code by the client in the absence of the indication from the application component that the work is complete; and

destroying the state of the application component in response to the indication from the application component to the operating service at the provided interface that the work is complete and without an indication from the client allowing the state of the application component to be destroyed.

26. (Previously added) The method of claim 25, further comprising:

subsequent to the destroying step, restoring the state of the application component in main memory upon receiving a method invocation from the client.

27. (Previously added) The method of claim 26, further comprising:

subsequent to the restoring step, destroying the state of the application component in response to the indication from the application component to the operating service at the provided interface that the work is complete and without an indication from the client allowing the state of the application component to be destroyed.

28. (Previously added) The method of claim 25, wherein the destroying step does not include notifying the client that the state of the application component is destroyed.